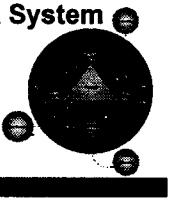


Mission Data System
HDCP Testbed

May 29-30, 2002

DRAFT

MDS HDCP Contacts

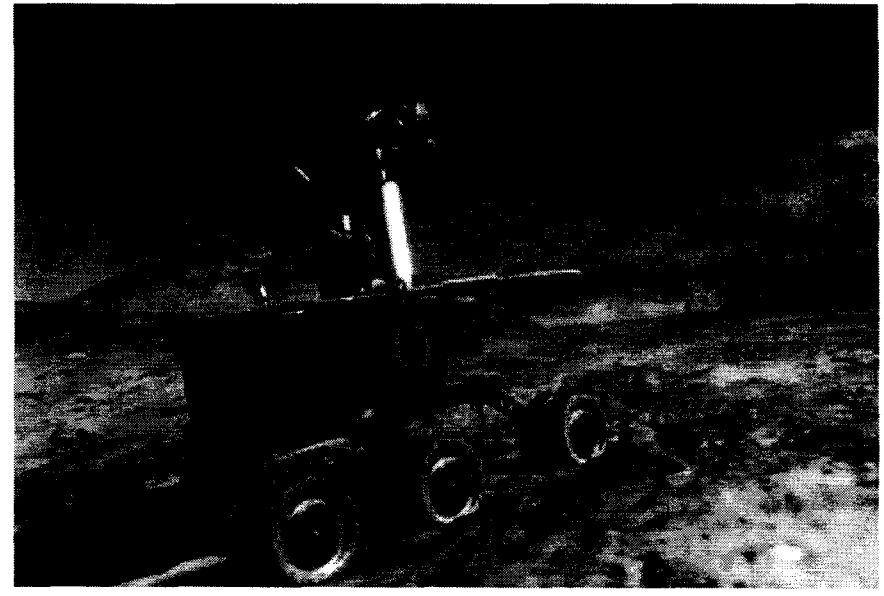
Dan Dvorak Daniel.L.Dvorak@jpl.nasa.gov	HDCP: Principle Investigator MDS: Deputy Architect
John Lai John.Y.Lai@jpl.nasa.gov	HDCP: Project manager MDS: Project manager
Kenny Meyer Kenny.Meyer@jpl.nasa.gov	HDCP: Liaison MDS: External partnerships
Kirk Reinholtz William.K.Reinholtz@jpl.nasa.gov	HDCP: Principal Investigator MDS: Chief Programmer

Introduction to MDS

Problem Domain

Autonomous control of physical systems

- Developed for unmanned space science missions involving spacecraft, landers, rovers, and ground systems
- Broadly applicable to mobile and immobile robots that operate autonomously to achieve goals specified by humans
- Architecturally suited for complex systems where "everything affects everything"



Approach

Product line practice to exploit commonalities:

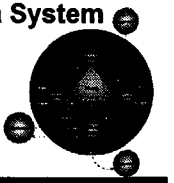
- Define a reference architecture to which missions/products conform
- Provide framework software to be used and adapted
- Define processes for systems engineering and iterative software development

Dependability Opportunities

- Systems Engineering Analysis & Design
  - Representations and tools to ensure methodical coverage, iterative refinement leading to higher fidelity designs
- Build & Test
  - Architectural correctness, modeling of complex interactions, model validation, COTS suitability, hardware/software trade space, predictability of schedule, ...
- Runtime Characteristics
  - Durability, diagnosability, quality of service guarantees, ...
- Mission Operations
  - Ease of error-free use, command verifiability, controllable level of autonomy, diagnosability, scalability, ...

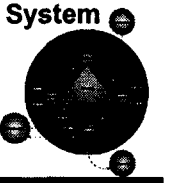
MDS Transition Path to Flight

- Mars Smart Lander ('09) Technology Infusion
  - Baselined MDS technology
    - System engineering
    - Software frameworks
  - Technology Selection process
    - RFI Technology Workshop 6/2002
    - Technology Selection 10/2002
    - Concept Review 6/2003
  - Sample technology categories
    - Software architecture with infused technologies
      - Covers end-to-end, cradle to grave, flight, ground and simulation
    - MDS-based cost model and estimate including system engineering, software adaptation and autonomy validation for MSL mission
    - Margin plan and assessment with respect to MSL avionics



MDS on RTSJ collaboration

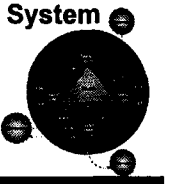
- MDS is currently implemented in C++
- Sun Lab/JPL collaborative to implement RT Java version of MDS
- Gosling and Bollella named JPL Distinguished Visiting Scientists



MDS Dependability Research Opportunities
Build/Test time Categories

Build/Test dependability categories address concerns associated with system definition, project management, implementation or verification.

Category	Description	MDS technical approach	Possible Measures and techniques
Architectural correctness of implementation	How well does the system's implementation reflect the analysis and design? How accurate is the translation from System Engineering to Software Engineering?	State analysis provides a system analysis and design methodology Component architecture provides rigorous method of composing software	• Percent of erroneous component & connector specifications
Modeling of complex interactions	Does the system provide a suitable means of expressing interactions? Do a certain types of defects map to certain type of software architectures? Run against different defect classes.	State analysis provides a system analysis and design methodology that exposes complex interactions between subsystems MDS provides a model-driven architecture that make adaptation easier	• Rate different defect class against different architectures
Model correctness	Is there a suitable separation of concerns to express physical models independently from information models? What's the right level of model fidelity for a particular application? How well does a model capture physical behavior?	MDS architecture provides for a disciplined use of models: structural, state effects, measurement, and command effects models.	• TBD
Architecture suitability	Are some architectures better suited to certain business cases?	MDS's emphasis on state and the management of physical interactions is well suited to resource-constrained systems.	• Measure design and development effort needed to accommodate new requirements.
COTS suitability	Is a real-time Java implementation suitable for a flight system? Are COTS products robust or efficient enough for use on the target system? How well does a COTS product scale to a real problem? What are the integration and process costs associated with incorporating a new product?	N/A	• TBD
Predictability of schedule and budget	How good is the team at meeting budget and schedule?	MDS has an iterative/incremental development methodology with clear exit points for collection of objective data.	• Earned value • Process feedback measures
Quality and defect reduction	Is the quality of the product improving? How do you know when the product is done?	MDS has an iterative/incremental development methodology with clear exit points for collection of objective data.	• COQUALMO • Defect seeding
Trade-space expressiveness	How do you establish criteria in the hardware/software trade space? *(performance vs flexibility)* Information sharing trade space? *(security vs safety)* System degradation trade space? *(survivability vs quality of service)*		• TBD



MDS Dependability Research Opportunities
Runtime Categories

Runtime dependability categories describe how well the system runs.

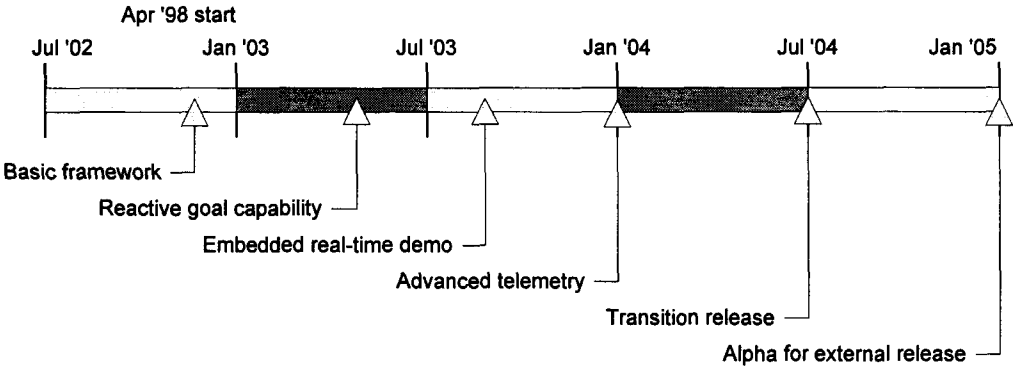
Category	Description	MDS technical approach	Possible Measures
Durability	How tolerant is the system to environmental variation? Does the system meet is up-time criteria? How do partial failures affect the ability of the system to meet mission objectives? Can the system be reliably upgraded using COTS capabilities like Java's dynamic loading?	Goal-driven operation permits highly tolerant success criteria. Partial failures handled at the lowest level possible, minimizing changes to goal network and thus to mission objectives.	• Accomplishment of highest-priority goals in the face of unexpected conditions.
Diagnosability	How easy is it to identify the cause of a fault? Is the system prone to a particular kind of fault?	MDS defines integral fault protection interfaces, allowing for wide range of detection & diagnosis techniques.	• Percent of false positives and false negatives during scenario-based testing
Quality of service guarantees	How accurately does the systems measure its state? How efficient is the system at doing the work for which it was designed?	State determination is a key architectural focus.	• Precision and delay of estimated state vs. true state.

MDS Dependability Research Opportunities
Operation Categories

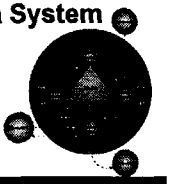
OperationTime dependability categories that describe how easy the system is to operate correctly.

Category	Description	MDS technical approach	Possible Measures
Diagnosability	How easy is it for an operator to find the cause of a system fault?	Device health reported in health state variables.	• Measure how well a fault is localized to a specific failure mode of a specific unit.
Ease of error-free use	How easily can operators instruct the system? How much effort goes into avoid system damaging mistakes	Goals specify *what*, not *how*. Goal net elaboration takes system-level interactions into account.	• Time to specify and validate a goal net versus a command sequence.
Command Verifiability	How much effort is needed to assure that the commands reflect intent? How easy is it to find command errors?	This is standard control law validation. MDS captures and reports command histories.	• TBD
Level of security	How immune is the system from malicious behavior?	MDS is designed for a non-malicious community.	• TBD
Level of Autonomy	Does the system provide autonomous capabilities that simplify operations? Can the system be customize to trade ease of development against ease of use?	Goal-driven operation intrinsically supports autonomy. The extent of automated goal elaboration trades ease-of-use against ease-of-development.	• Measure operational load for goal-based vs. command-based operations.
Maintainability	How easy is it to maintain the system?	Explicit information in state, measurement, and command histories, as well as an event log, facilitate maintenance.	• Measure how long it takes to detect and fix seeded defects.
Scalability	How easy is it to scale the system?	Explicit representation of states and modeling of interactions encourage confidence.	• Measure architectural variation as a system evolves toward high-fidelity behavior.

MDS Schedule

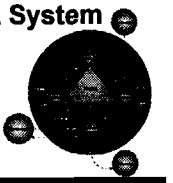


Date	Theme	Scope
10/30/02	Basic Framework	• Basic set of frameworks sufficient to build flight, ground and test deployments with goal based commanding mechanism and state based functional layer with h/w interface, estimation, control and data management • EDL example for the powered terminal descent phase • Enhanced user documentation
4/30/03	Reactive Goal Capability	• Framework for resource Management (both planning & execution) • Framework for fault detection, with recovery by the goal layer • Frameworks providing improved reliability and with more user documentation • Adaptation example for end-to-end EDL with ballistic hypersonic entry
8/31/03	Embedded Real Time Demonstration	• Performance enhancement of the MPE framework algorithms for elaboration and planning • Framework performance testing in embedded environment with realistic adaptation examples - Surface system mobility functions, including turn-in-place and drive-to-location, in a target rover processor- Flight deployment for full end-to-end EDL, including hypersonic entry with roll control
12/31/03	Advanced Telemetry	• Framework policy mechanisms for high data volume scenarios-EDL control algorithms and frameworks to support Lidar/radar interface for terrain mapping and safe site selection • Maneuver to safe site based on Lidar/radar measurements
6/30/04	Transition Release	• Framework optimization with enhanced documentation in preparation for external release • Adaptation example of rover surface system with hazard avoidance, executing in a target processor real-time environment (FIDO HW)
12/31/04	Alpha for External Release	• Tested frameworks, adaptation examples and user documentation ready for external release



MDS Release to HDCP

- MDS Release 2 will be made available to HDCP ~30 days after the JPL delivery
- Products
  - Architecture Design Document
  - State Analysis Document
  - Framework Descriptions
  - Source code
  - Adaptation Examples
  - Adaptation Guides
  - Release Description Documents



Mars Smart Lander Infusion Opportunities

- Expected capability list
- Deadlines
  - RFI Technology Workshop 6/2002
  - Technology Selection (In-gate) 10/2002
  - Concept Review (CR) 6/2003
    - MDS-based cost model and estimate including system engineering, software adaptation and autonomy validation for MSL mission
    - Margin plan and assessment with respect to MSL avionics
    - Software architecture with infused technologies
      - End-to-end flight, ground and simulation
    - MSL system engineering/state analysis process and results captured in DOORS and SDS tools
    - Risk list identified and prioritization
    - Relationship to Release 5 software products
      - Mobility in workstation-base simulated and physical rover (Vxworks)
      - Integrated MPE with Elaboration, Scheduler, GEL as U/L product
      - Telemetry for MPE as data product
      - Policy for data management for different products
      - Policy on data transport (not CFDP)
      - Enhanced SDS and Elaboration tool